Problem 5 (6 Points)

Now you will implement a wrapper method. This will iteratively determine which features should be most beneficial for predicting the output. Once more, we will use the MTCars dataset predicting `mpg`.

In [1]:
```
import numpy as np
np.set_printoptions(precision=3)
from sklearn.svm import SVR
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import train_test_split
import itertools

feature_names = ["mpg","cyl","disp","hp","drat","wt","qsec","vs","am","gear","carb"]
data = np.array([[21,6,160,110,3.9,2.62,16.46,0,1,4,4], [21,6,160,110,3.9,2.875,17.02,0,1,4,4], [22.8,4,108,93,3.85,2.3
                [18.1,6,225,105,2.76,3.46,20.22,1,0,3,1], [14.3,8,360,245,3.21,3.57,15.84,0,0,3,4], [24.4,4,146.7,62,3.
                [17.8,6,167.6,123,3.92,3.44,18.9,1,0,4,4],[16.4,8,275.8,180,3.07,4.07,17.4,0,0,3,3],[17.3,8,275.8,180,3
                [10.4,8,460,215,3,5.424,17.82,0,0,3,4],[14.7,8,440,230,3.23,5.345,17.42,0,0,3,4],[32.4,4,78.7,66,4.08,2
                [21.5,4,120.1,97,3.7,2.465,20.01,1,0,3,1],[15.5,8,318,150,2.76,3.52,16.87,0,0,3,2],[15.2,8,304,150,3.15
                [27.3,4,79,66,4.08,1.935,18.9,1,1,4,1],[26,4,120.3,91,4.43,2.14,16.7,0,1,5,2],[30.4,4,95.1,113,3.77,1.5
                [15,8,301,335,3.54,3.57,14.6,0,1,5,8],[21.4,4,121,109,4.11,2.78,18.6,1,1,4,2]])
target_idx = 0
y = data[:,target_idx]
X = np.delete(data,target_idx,1)
```

Fitting a model

The following function is provided: `get_train_test_mse(X,y,feature_indices)`. This will train a model to fit the data, using only the features specified in `feature_indices`. A train and test MSE are computed and returned.

In [2]:
```
def get_train_test_mse(X, y, feature_indices=None):
    if feature_indices is not None:
        X = X[:,feature_indices]
    X_tr, X_te, y_tr, y_te = train_test_split(X,y,random_state=12,train_size=int(len(y)*.8))
    model = SVR()
    model.fit(X_tr,y_tr)
    mse_train = mean_squared_error(y_tr,model.predict(X_tr))
    mse_test = mean_squared_error(y_te,model.predict(X_te))
    return mse_train, mse_test

mse_train, mse_test = get_train_test_mse(X, y, None)
print(f"Model using all features:    Train MSE={mse_train:.1f},  Test MSE={mse_test:.1f}")
```

```
Model using all features:    Train MSE=16.1,  Test MSE=18.3
```

Wrapper method

Now your job is to write a function `get_next_pair(X, y, current_indices)` that considers all pairs of features to add to the model.
`X` and `y` contain the full input and output arrays. `current_indices` lists the indices currently used by your model and you want to determine the indices of the 2 features that best improve the model (gives the lowest test MSE). Return the indices as an array.

If you want to avoid a double for-loop, `itertools.combinations()` can help generate all pairs of indices from a given array.

In [3]:

```
def get_next_pair(X, y, current_indices):
    # YOUR CODE GOES HERE

    best_indices = None
    lowest_mse = float('inf')
    ind = []

    for i in range(X.shape[1]):
        if i not in current_indices:
            ind.append(i)
    pairs = itertools.combinations(ind,2)

    for pair in pairs:
        indices = list(current_indices)+list(pair)
        indices = np.array(indices, dtype = int)
        mse_train,mse_test = get_train_test_mse(X,y,indices)

        if (mse_test < lowest_mse):
            best_indices = pair
            lowest_mse = mse_test

    return best_indices
```

Trying out the wrapper method

Now, let's start with an empty array of indices and add 2 features at a time to the model. Repeat this until there are 8 features considered. Each pair is printed as it is added.

The first few pairs should be:

- (2, 5)
- (0, 8)

In [4]:

```
indices = np.array([])
while len(indices) < 8:
    pair = get_next_pair(X, y, indices)
    print(f"Adding pair {pair}")
    indices = np.union1d(indices, pair)
```

```
Adding pair (2, 5)
Adding pair (0, 8)
Adding pair (6, 7)
Adding pair (4, 9)
```

Question

Which 2 feature indices were deemed "least important" by this wrapper method?

The least important pair by this wrapper method is (1,3)

In [ ]:

```

```